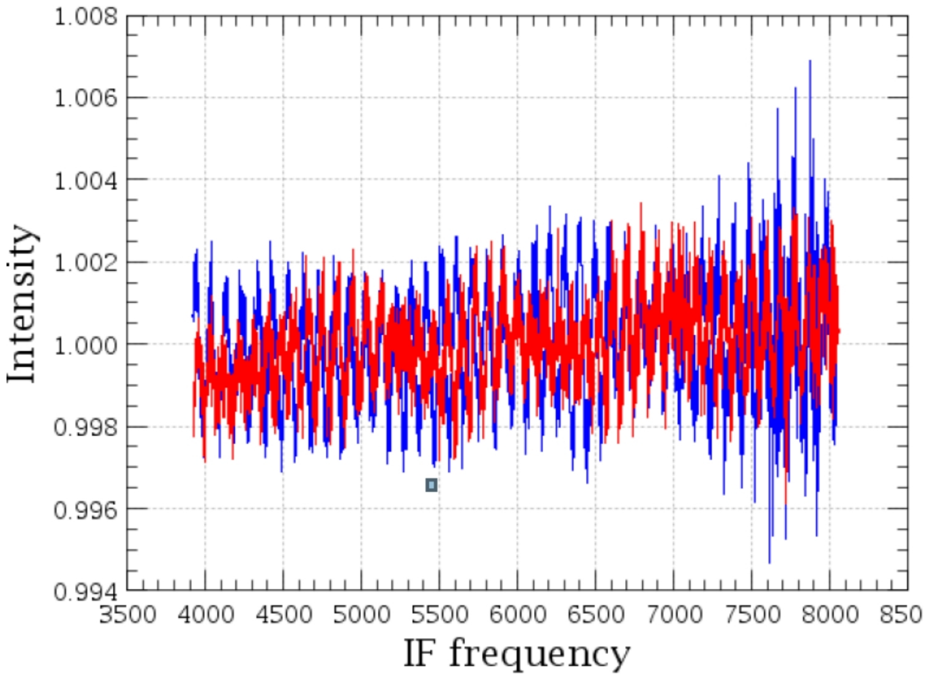
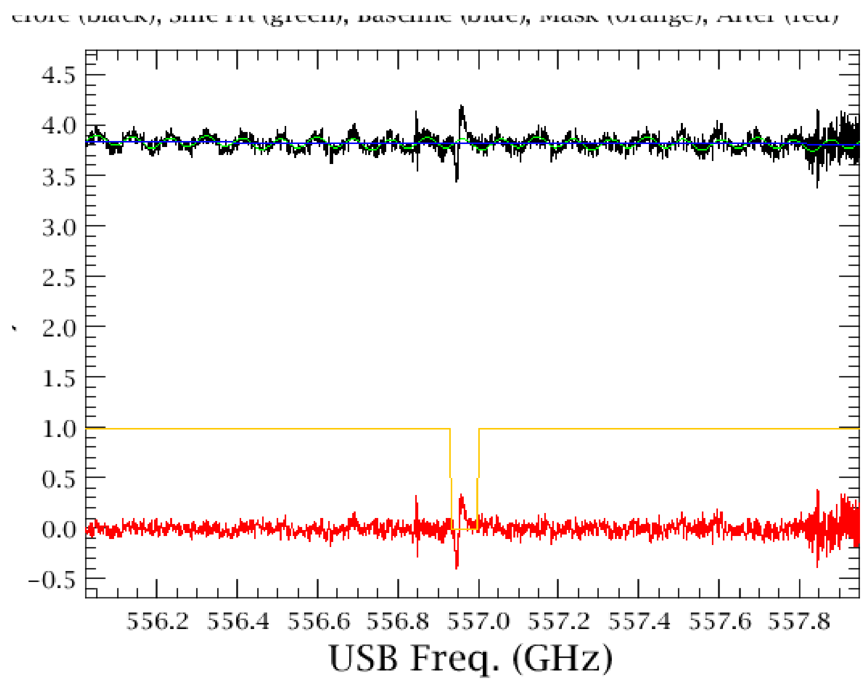
Removing HIFI Baseline Artifacts

Adwin Boogert
NHSC/IPAC, Caltech





Baseline Artifacts

Similar to ground-based heterodyne instruments, HIFI instabilities and imperfect AOT or pipeline design leave instrumental signatures in Level 2 and 2.5 data:

- Standing waves with different periods, shapes, amplitudes
- Slopes
- Curvatures
- Offsets
- Jumps between sub-bands

HIPE offers several tools to correct for this:

1) FitHifiFringe: generic HIFI-optimized sine-wave fitting tool
2) DoFilterloads: optional pipeline step to remove particular load waves
3) HEBStWvCatalogCorrection.py: advanced user script for band 6+7 HEB electronic waves
4) FitBaseline: generic HIFI-optimized polynomial fitting tool

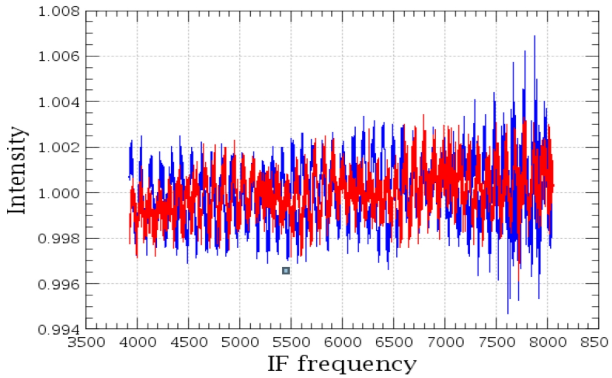
Baseline Problems: Standing Waves

Standing waves produced by optical and electronic components. Pipeline removes them by subtracting reference sky or load spectra.

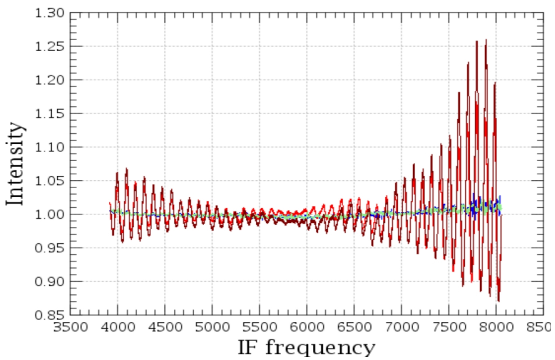
Standing wave residuals sometimes seen in Level 2 data. Strength in agreement with HSPOT predictions.

Wave-type is HIFI-band dependent:

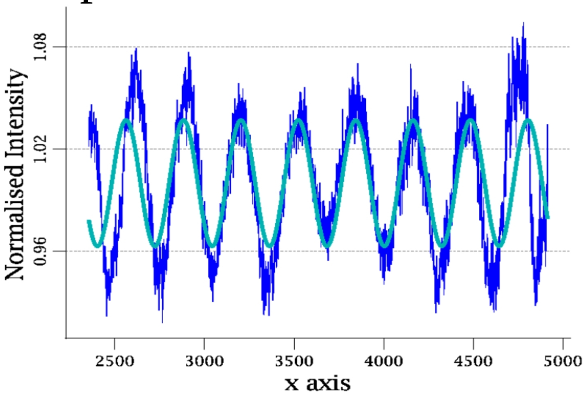
- Beamsplitter bands 1, 2, and 5 show sine waves



- Diplexer bands 3 and 4 show sine waves with amplitude increasing to IF band edges



- HEB bands 6 and 7 waves are not sine waves. Requires special treatment.



Optical Waves Observed in Level 2 Data

Origin (Mixer to..)	Period [MHz]	Amplitude Bands 1-2	Amplitude Bands 3-4	Amplitude Band 5	Amplitude Bands 6-7
Cold Black Body	98	3-4%	1-2%	1%	<1%
Hot Black Body	92	3-4%	1-2%	1%	<1%
Local Oscillator	100	<1%	2-4%	3%	3-25%
Roof Top Mirror	620	n.a.	1-2%	n.a.	<1%

In this workshop we will discuss ways to remove the residual waves using FitHifiFringe.

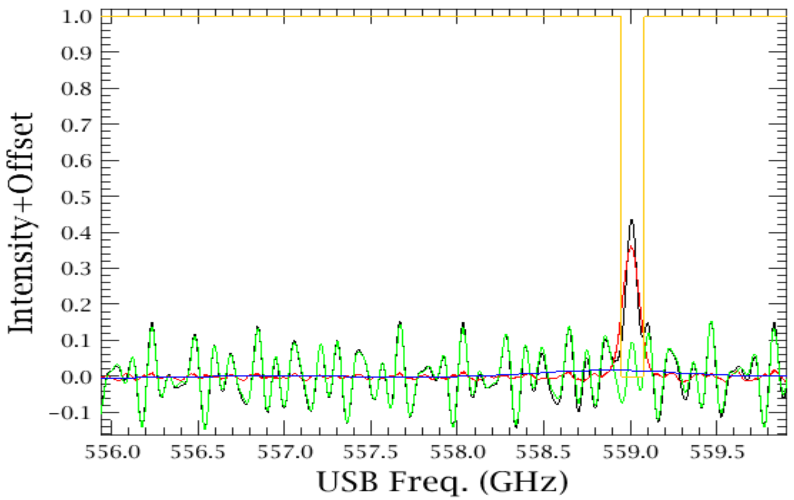
Alternatively, the Cold and Hot Black Body load waves can be removed with the DoFilterloads pipeline step or in combination with FitHifiFringe.

Bands 6 and 7 also have a strong electronic standing wave with a period of 320 MHz. The special "current match method" to remove this waves is still under development. We will organize a dedicated webinar now that this advanced user script is available in HIPE 9.1.

Standing Wave Removal with Sine Wave Fits: *FitHifiFringe*

•In order to fit sine waves to the standing waves, the standing waves need to be separated from:

– other baseline fluctuations
– emission/absorption lines



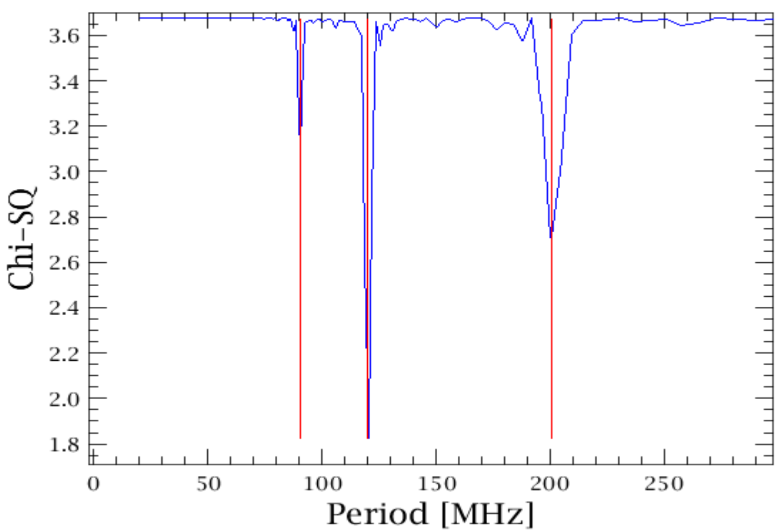
•Fit N sine waves with different periods, amplitudes, phases to baseline-subtracted, line-masked spectrum.

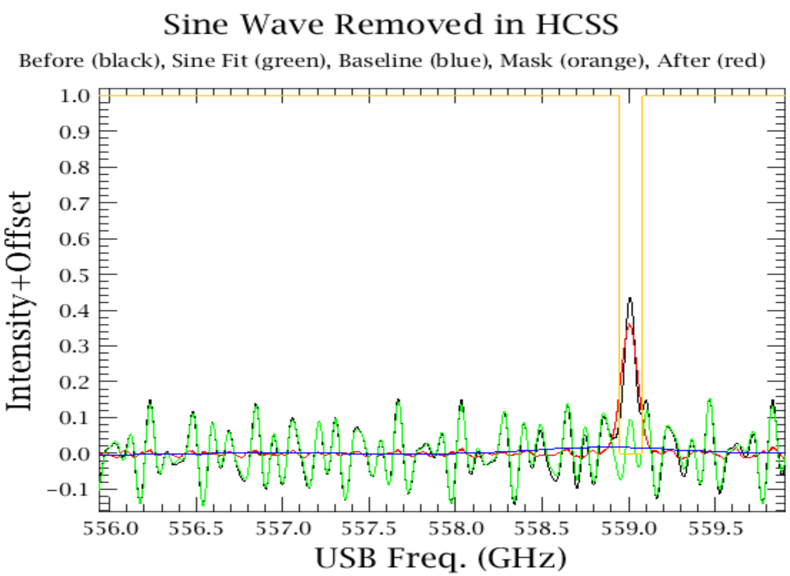
•Subtract SW fit from original input data. FitHifiFringe does this by default automatically. But it is not always perfect, and the user has to inspect each plot and may need to set line masks by hand.

Standing Wave Removal with FitHifiFringe

FitHifiFringe shows fitted periods, amplitudes, phases in HIPE console, stores them in ObsContext, and by default produces 2 plots:

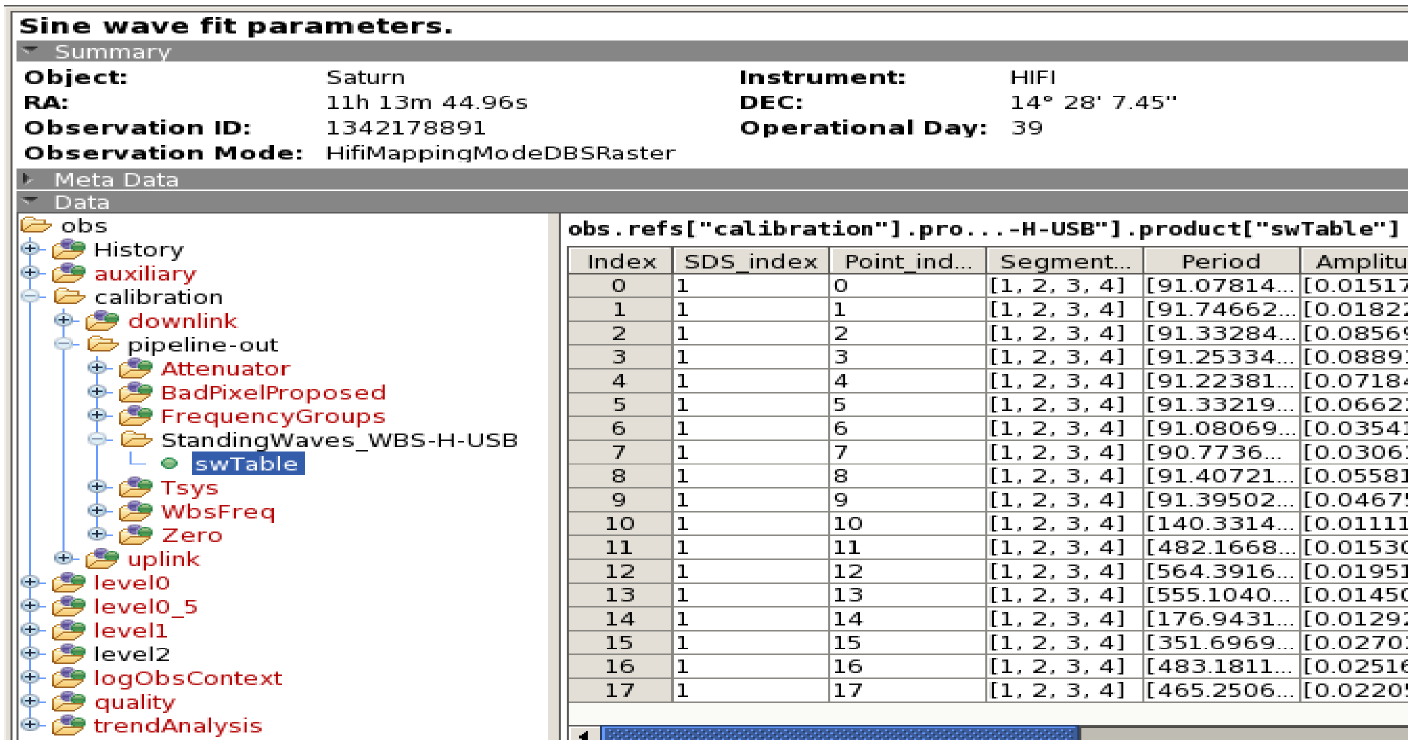
- χ^2 as function of period. Minima found are indicated with red vertical lines
- Result plot: input data, sine wave fit, baseline, mask, sine-wave subtracted spectrum





Standing Wave Removal with FitHifiFringe

best sine wave fit parameters stored in Observation Context table



Sine wave fit parameters.

Summary

Object:	Saturn	**Instrument:**	HIFI
RA:	11h 13m 44.96s	**DEC:**	14° 28' 7.45"
Observation ID:	1342178891	**Operational Day:**	39
Observation Mode:	HifiMappingModeDBSRaster		

Meta Data

Data

obs
History
auxiliary
calibration
downlink
pipeline-out
Attenuator
BadPixelProposed
FrequencyGroups
StandingWaves_WBS-H-USB
swTable
Tsys
WbsFreq
Zero
uplink
level0
level0_5
level1
level2
logObsContext
quality
trendAnalysis

obs.refs["calibration"].pro...-H-USB"].product["swTable"]

Index	SDS_index	Point_ind...	Segment...	Period	Amplitu
0	1	0	[1, 2, 3, 4]	[91.07814...	[0.01517
1	1	1	[1, 2, 3, 4]	[91.74662...	[0.0182
2	1	2	[1, 2, 3, 4]	[91.33284...	[0.0856
3	1	3	[1, 2, 3, 4]	[91.25334...	[0.0889
4	1	4	[1, 2, 3, 4]	[91.22381...	[0.0718
5	1	5	[1, 2, 3, 4]	[91.33219...	[0.0662
6	1	6	[1, 2, 3, 4]	[91.08069...	[0.0354
7	1	7	[1, 2, 3, 4]	[90.7736...	[0.0306
8	1	8	[1, 2, 3, 4]	[91.40721...	[0.0558
9	1	9	[1, 2, 3, 4]	[91.39502...	[0.0467
10	1	10	[1, 2, 3, 4]	[140.3314...	[0.01111
11	1	11	[1, 2, 3, 4]	[482.1668...	[0.0153
12	1	12	[1, 2, 3, 4]	[564.3916...	[0.0195
13	1	13	[1, 2, 3, 4]	[555.1040...	[0.0145
14	1	14	[1, 2, 3, 4]	[176.9431...	[0.0129
15	1	15	[1, 2, 3, 4]	[351.6969...	[0.0270
16	1	16	[1, 2, 3, 4]	[483.1811...	[0.0251
17	1	17	[1, 2, 3, 4]	[465.2506...	[0.0220

Standing Wave Removal with FitHifiFringe: *Limitations*

Standing waves were successfully removed with FitHifiFringe in all bands, but it cannot be guaranteed for every observation:

- If there are so many lines, that little 'clean' baseline is left
- Bands 3 and 4 'diplexer' waves are not pure sine waves. It helps if lines are near middle of band, where amplitudes are lowest.
- Band 6 and 7 'electronic' waves are not pure sine waves. An alternative 'pattern-matching' method is available in the advanced HEBStWvCatalogCorrection.py script.
- In specific cases, esp. for deep integrations in bands 1 and 2, the doFilterLoads method mentioned in next slides is better alternative

Removing CBB and HBB Waves

The 92 and 98 MHz hot and cold load waves are divided into the sky spectra during passband calibration:

$J_{ON}-J_{OFF}=(C_{ON}-C_{OFF})/(C_{HBB}-C_{CBB})*(J_{HOT}-J_{COLD})$

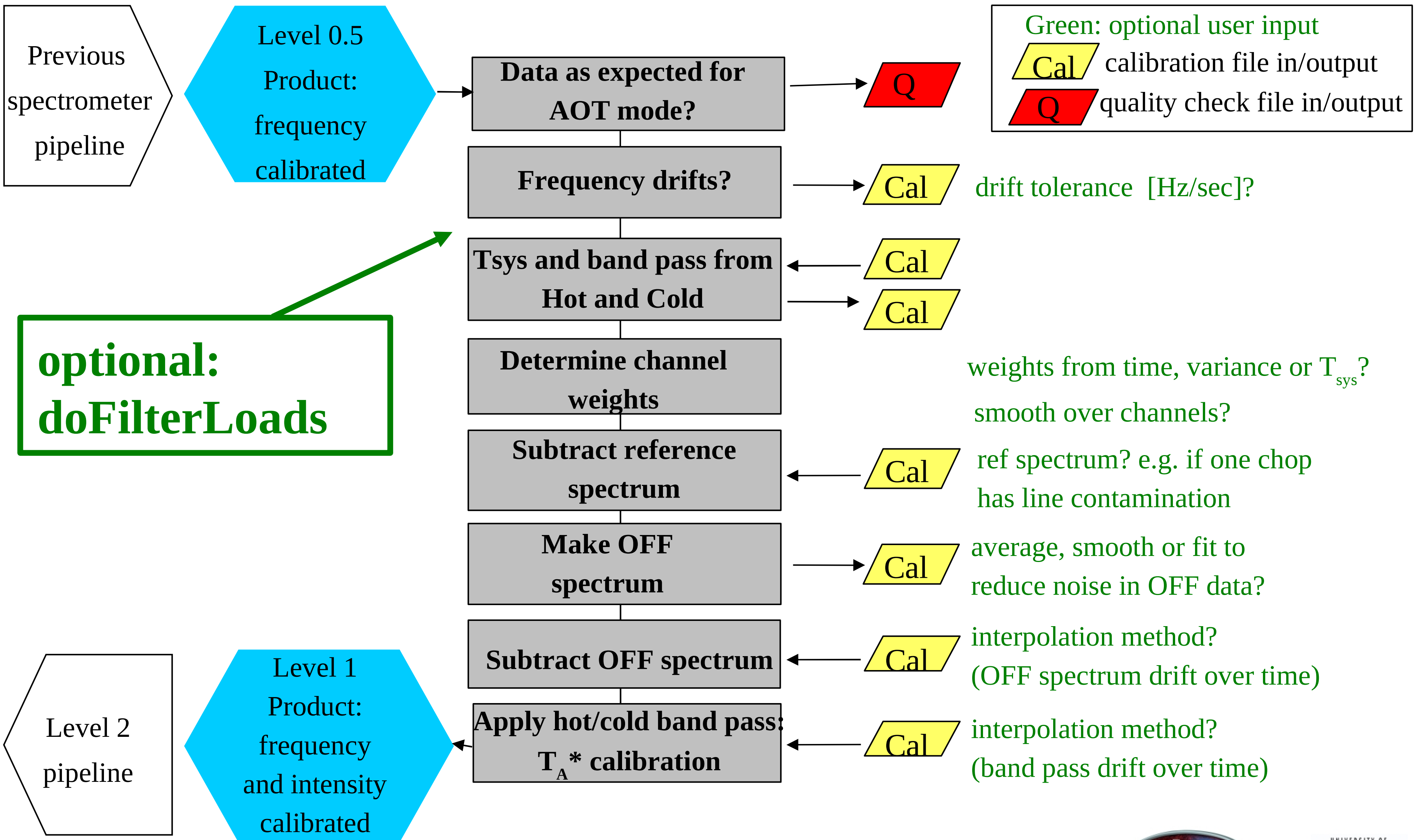
In HIPE 8 a new pipeline option doFilterLoads was introduced that removes the 92 and 98 MHz waves.

To isolate these waves from the other waves (most importantly the 100 MHz LO-mixer wave), the load spectra are divided by sky spectra.

DoFilterLoads works as follows:

- Compute C_{HBB}/C_{OFF} and C_{CBB}/C_{OFF}
- Remove waves using cubic spline or FFT
- Multiply C_{OFF} back in to get modified C_{HBB} and C_{CBB}

Generic Pipeline (Level 0.5 -> 1)



Removing CBB and HBB Waves

Advantage FilterLoads method over sine wave fitting level 2 spectra:

- Load spectra not 'contaminated' by emission or absorption lines, making the method more reliable and objective.

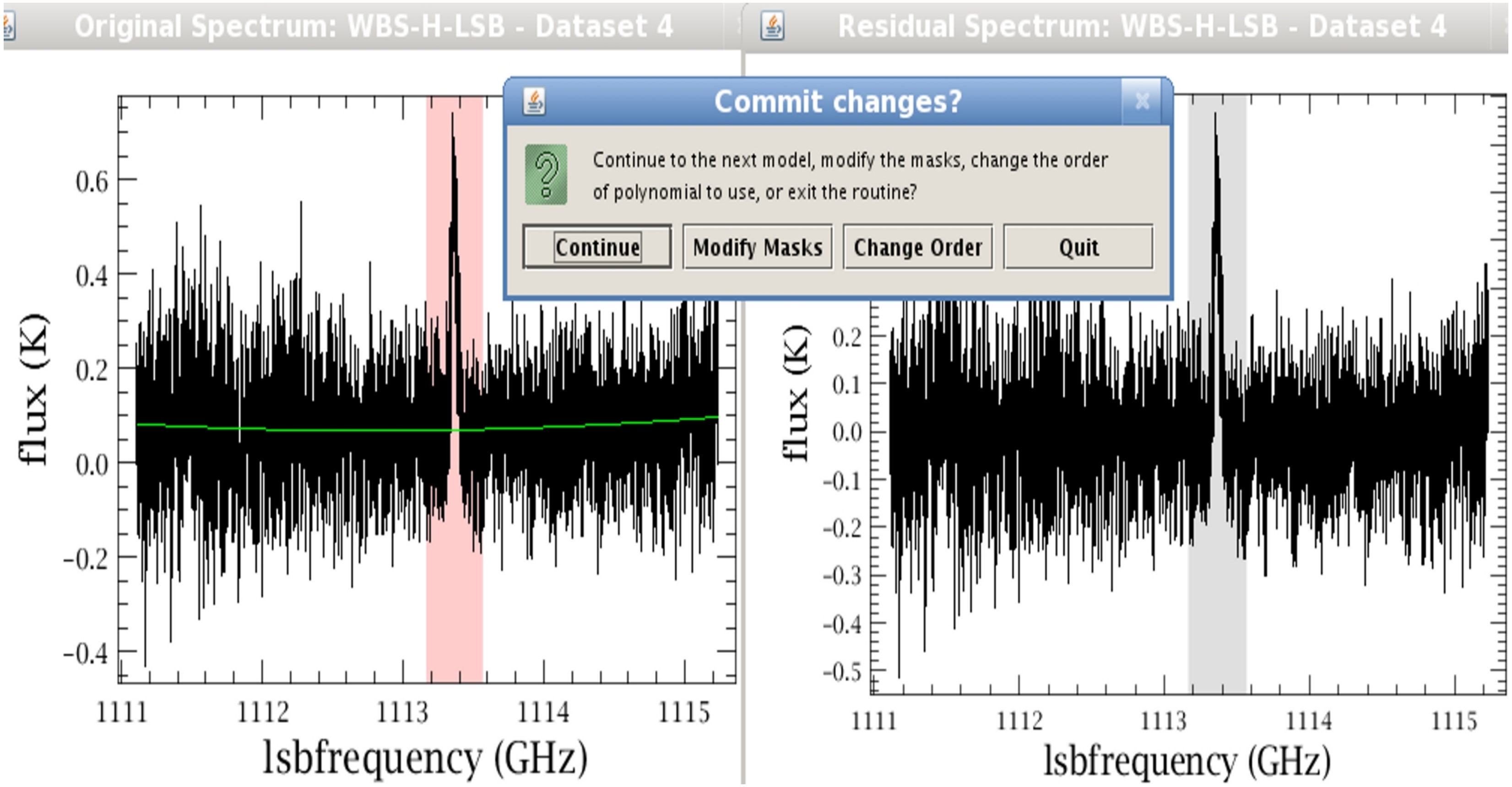
Caveats and side effects:

- Method only works if off sky measurements available.
- Outer edges of sub-bands will still show artifacts because of smoothing edge effects
- Method only works if load waves stronger than 100 MHz LO-mixer wave, i.e.,
  - for bands 1 and 2
  - Central part bands 3 and 4 (LO-mixer wave dominates in the outer parts)
  - Little effect on band 5-7

Baseline Correction with FitBaseline

- FitBaseline: user-friendly tool for polynomial baseline fitting and subtraction or division
- Developed by team of CHESS 'KP students' (Mihkel Kama, Tom Bell, Sandrine Bottinelli—**thank you all!**) and ICC members (Caux, Rabois, Xie, Boogert)
- Features include:
  - Mask lines or spurs by clicking or use automated masking procedure
  - Disable mask, change polynomial order iteratively by user
  - In SScans, automatically determine sideband of line, and then propagate masks to other LO settings
  - Works for all observing modes
  - Line mask is stored in table in ObsContext
  - Before/after spectra and mask spectra are all stored
  - Re-do fits using stored masks and polynomial orders. Useful after new pipeline processing.

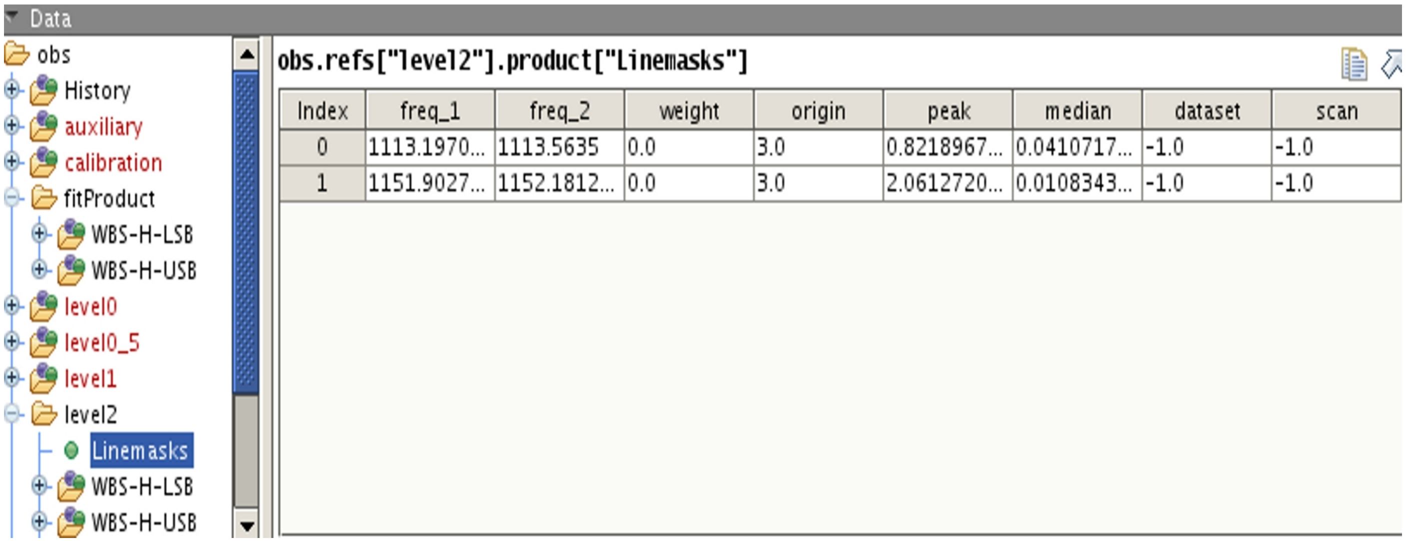
Baseline Removal with Polynomial Fits: *FitBaseline*



For polynomial baseline fitting with FitBaseline one may need to inspect each plot and adjust the line masks by hand.

Baseline Correction with FitBaseline

Line Mask



Masked frequency ranges stored in a table in the ObsContext.
This is in fact a line list.

Efficient Baseline Correction in HIPE

Typical HIFI post-pipeline data flow:

1. Data inspection, finding spurs, modifying flags
2. FitHifiFringe to remove standing waves
3. FitBaseline to remove baseline offsets and slopes
4. Cube construction or deconvolution

Steps 2 and 3 can be A LOT of interactive work. The next slide shows a way to do it more efficiently.

Efficient Baseline Correction in HIPE

- In the demos we will show ways to limit the work using various options in FitBaseline and FitHifiFringe.
- Another avenue became available with the introduction of FlagTool in HIPE 8.0. Flag lines and bad data once and for all.
  - Line flags will be picked up by other tools, so mask inspection in FitBaseline and FitHifiFringe no longer needed.
  - Save Linemask table produced by FlagTool to disk and use it when data has been re-pipelined.
  - Caveat: **FlagTool is still being improved!**
  - See demo

EXTRA SLIDES

Level 1 → 2 Processing
Standing Wave Removal: Principles

- Baseline and mask determined using iterative filtering, smoothing, sigma-clipping algorithm. Needs to know 'typical' longest standing wave period, to separate waves from other baseline structure.
- N sine waves A*sin(x)+B*cos(x)=C*sin(x+phase) fitted to baseline-subtracted, line-masked spectra iteratively:

A) Determine period, amplitude, phase of wave 1 at minimum chi^2

B) Determine period wave 2 at minimum chi^2 in sine-wave subtracted data

C) Solve for amplitude and phase ***combined waves 1+2*** using 'LU' matrix decomposition in original input data

D) Subtract ***waves 1+2*** from original input data

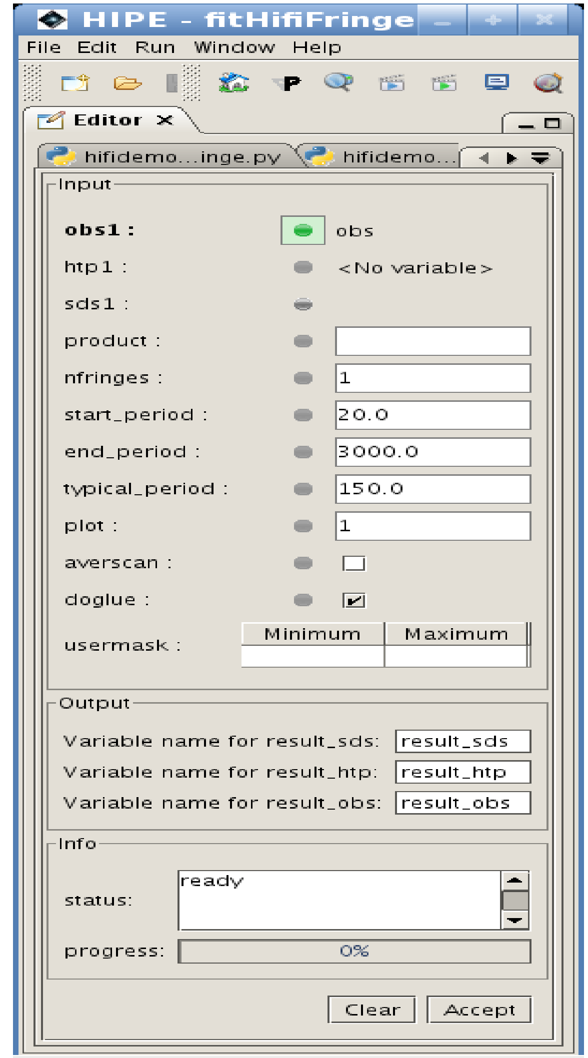
E) Repeat steps B, C, D for waves 1+2+3, ..., 1+2+3+...+N

Standing Wave Removal with FitHifiFringe

FitHifiFringe input:

- ObservationContext, HifiTimelineProduct, SpectrumDataset, or simplespectrum
- product for ObsContext (e.g. WBS-H-USB)
- nfringes: number of sine waves to fit
- start_period: shortest period SW to search for
- end_period: longest period SW to search for
- typical_period: typical SW expected in data. Longer period structures are assumed to be baseline or sky features.
- plot: 0=not, 1=most important plots
- doglue: determine SW on combined sub-bands
- usermask: user-defined mask
- sub_base: subtract smooth baseline as well
- averscan: determine SW from average of all scans, and subtract that from all. Deprecated?

FitHifiFringe output: sine wave(s)-subtracted data (obs, htp, sds) and list of sine wave parameters fitted



Removing CBB and HBB Waves

HIPE demo:

- Re-run pipeline with doFilterLoads step on Mars observation of band 1B ('easy' case)
- Re-run pipeline with doFilterLoads step on Mars observations of band 3B ('tough' case—this is diplexer band)
- Remove residuals with sine wave fitting (fitHifiFringe task)
- Compare doFilterLoads with fitHifiFringe results